DESIGNING CONVERSATIONAL INTERFACES WITH MULTIMODAL INTERACTION

Josh Bers, Scott Miller, John Makhoul
BBN Technologies
70 Fawcett St.
Cambridge, MA 02138

ABSTRACT

Our current research focuses on developing conversational interfaces to on-line applications through speech recognition technology. We have developed a prototype system that combines pen and speech input from the on-line user in a web-browser. VoiceLog is a voice-enabled connection to a web-server that allows one to obtain vehicle diagrams and to place orders for specific parts in these diagrams. VoiceLog features a novel client-server approach to speech recognition, modular reusable components and a simple Java-based interface. This paper briefly describes the system and its architecture including the handling of simultaneous input from pen and speech, the production of audio and visual feedback, and the management of multimodal dialogue.

1. INTRODUCTION

Traditional keyboard and mouse interfaces are impractical on small portable devices. Spoken language systems, such as voice enabled browsing, offer an intuitive way of accessing the growing amount of on-line information. The next generation of mobile networked users will use speech and gesture to enter and retrieve information.

Recent studies have shown the benefits of integrating multiple input modalities, i.e. buttons, pen, and speech, for graphical and audio-only applications on hand-held devices [1, 2].

We have developed an interface that enables the user to access and to enter data on the World Wide Web using voice and a stylus. The system runs on a mobile pen-based computer with a microphone and a wireless connection to the Internet.

2. VOICELOG SYSTEM

Similar to previous multimodal systems (e.g., [3, 4, 5]) the user of VoiceLog selects items on the display with the pen or with speech and specifies actions verbally. A sample of an interaction with the system follows:

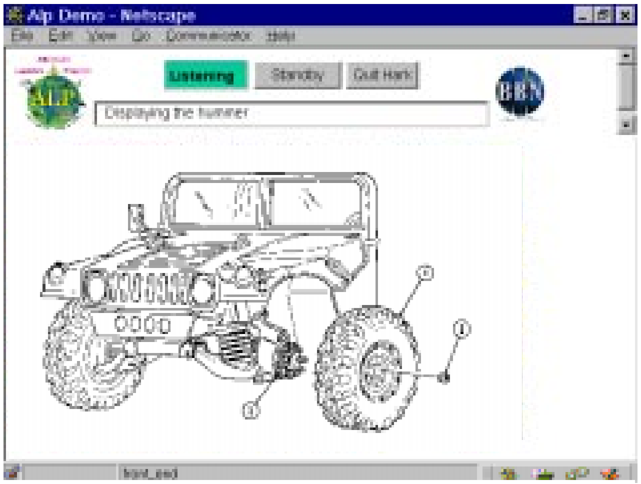
(U stands for user input; VL stands for VoiceLog response)

U: “Show me the humvee.” [humvee is a nickname for the HMMWV military jeep]

VL: displays an image of the HMMWV.

U: “Expand the engine.”

VL: flashes the area of the engine and replaces the image with a part diagram for the engine.



Displaying the hummer

Figure 1: VoiceLog’s interface displaying a part-diagram.

U: (while pointing at a region with the pen) “Expand this area.”

VL: highlights the selected area, the fuel pump, and displays an expanded view.

U: (points to a specific screw in the diagram) “I need 10.”

VL: brings up an order form filled out with the part name, part number and quantity fields for the item.

2.1 Interface

Our design goals for the interface were simplicity and efficiency of operation. We chose to implement VoiceLog as Java applets running in a web browser because of the ubiquity of the platform and the familiarity of the interface to many users. The resulting single page layout fits on the displays of most hand-held computers.

Figure 1 shows the VoiceLog web-page. The page is divided into two frames. The small upper frame, which is static, displays the system’s status and contains GUI buttons for controlling the speech recognizer. The larger lower frame displays either diagrams from the parts catalog or an order form. Catalog images contain “hot” regions corresponding to individual parts that may be selected with the pen or through speech. The order form allows multimodal form filling by pointing to a slot and either speaking a value or writing it in with the stylus.

Feedback is given in both audio and visual form. Objects show their names and flash their location on the image when pointed at and selected, respectively. Audio repair dialogue handles incomplete requests and speech recognition failures.

Voice enabled interfaces, like human ears, suffer from noisy channels. We use a small, ear-mounted, noise-canceling microphone made by Telex to keep the signal clean. The default click-to-talk mode, where one must touch the display with the pen before speaking, is suited to noisy environments. For quieter settings one may use the hands-free, continuous recognition mode, where the user can give multiple sequential commands without touching the screen.

Our mobile, wireless device is a Fujitsu Stylistic 1200 pen PC running Windows 95. We use a RangeLan2 PC card from Proxim for wireless network connectivity.

2.2 Architecture

The system's block diagram, as shown in Figure 2, consists of a client and a server communicating across a TCP/IP network. The speech recognition back-end and the web-server run on the server, while the client hosts the front-end of the speech recognition and a web-browser running Java code.

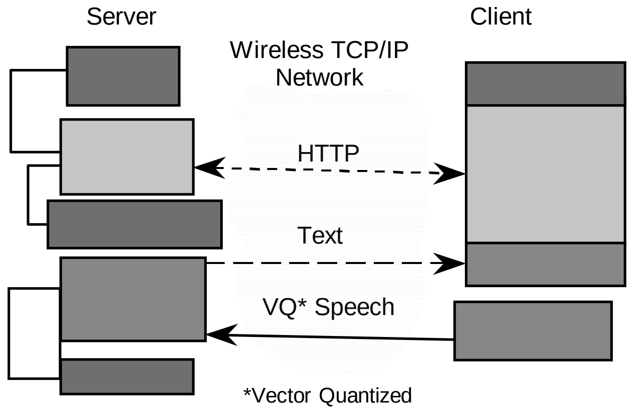


Figure 2: Block diagram of the networked architecture.

The client-side Java application has three major components: speech input, pen input, and integration and response (I&R). Each component runs in its own processing thread. Separate threads communicate through event queues and Java synchronized methods. Figure 3 details the client-side data flow.

In the following two sub-sections we describe the data flow and the management of the interaction with the user.

Data flow. A voice recognition system first processes speech input. We use the Hark recognizer, BBN's speaker-independent, continuous speech recognition system. In our architecture, we have split up the computational workload of speech recognition such that the client machine performs front-end signal processing and sends the vector-quantized data to the server for decoding into words. This design allows large vocabulary recognition on small devices because the compute and memory intensive processing is done on the server.

We have observed that a 9600-baud connection is sufficient for transmitting the speech data.

Pen input consists of stylus (mouse) motion as well as up and down (click) events.

The pen and speech threads parse pen gestures and speech recognition results, respectively, into event structures. The input threads then place the events onto a queue managed by the I&R thread (see Figure 2). The I&R thread reads the events from the queue and responds with the appropriate action.

As shown in Figure 2, feedback to user input is handled at two levels. The input threads handle responses that require no higher-level processing, e.g., when the pen enters a "hot" region in a diagram, the part name is displayed. Responses which may require integrating input from multiple modalities, e.g., "Show me this diagram," are handled by the I&R thread. This multi-layered feedback assures a quick response to user input [7].

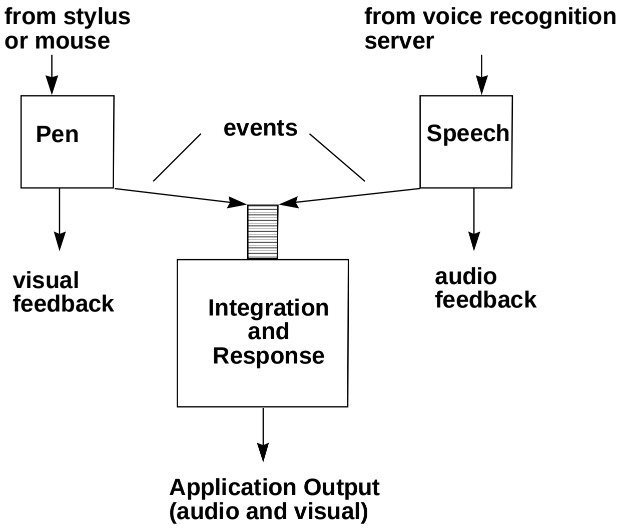


Figure 3: Data flow through the client application

Dialogue management. There are four major states of interaction handled by the I&R process: Idle - waiting for input; Object_input - object selected, waiting for command selection; Action_input - received a command, waiting for object selection; and Both_input - object and command specified, ready to take action, waiting for additional refinement of command.

Time-outs in each state maintain the dialogue flow. For example, if the user says, "order six wheels," the system will time-out in the both_input state and print the action to be taken by the system in the status window before performing the action, i.e., "ordering 6 wheels," and then displaying the order form. By adjusting the duration of these time-outs one can tailor the speed of the interaction to user preference and to the delay characteristics of the input modalities.

Two repair states handle failures to supply either an object or an action. Time-outs in the action_input and object_input states control entry into their respective repair states. For instance, if the user forgets to choose an item for an order command, after a time-out, the system will prompt orally, "order which object?" and display the incomplete command in the status window, "ordering 6 ___." Speech recognition of object names is then activated so that the user can give the repair with either verbal or pen input.

3. CONCLUSIONS

With the advent of portable/wearable computing devices, there has been an increased demand for computers in mobile, noisy environments, e.g., cars, marketplace, parking lots, retail showrooms, cocktail parties, etc. VoiceLog presents an intuitive multimodal interface for web-based data access and entry in mobile environments. Its unique design features:

- Modular multimodal architecture - facilitates reuse and enhancement of its components. We have reused the speech input thread in another voice-enabled application, GTNPhone [7].
- Centralized speech recognition server - permits large sized vocabularies on lightweight client machines.
- Simple single-window interface - places a low cognitive demand on the user.
- Web-based architecture - ensures a ubiquitous platform and simplifies system maintenance and upgrades. All application code and data including speech recognition grammars and vocabularies are maintained in one place, the web-server.

Currently the VoiceLog system is running on UNIX workstations, desktop PC's and a mobile, pen-based PC. Our future plans for the VoiceLog system are to leverage other technologies: speaker verification for order authorization and text to speech generation for more conversational interactions. Other future work will explore how the centralized speech recognition architecture scales with the number of simultaneous users. We would also like to perform user studies to evaluate our system and to compare it with traditional interfaces.

Acknowledgments

This work is supported by the Defense Advanced Research Projects Agency under contract DABT63-94-C-0061. Thanks to Rusty Bobrow, Lyn Bates, Marie Meteer, Rich Schwartz and Steve Grubb for their valuable contributions.

References

1. Oviatt, S. L., A. DeAngeli, and K. Kuhn, *Integration and synchronization of input modes during multimodal human-computer interaction*. Proceedings of CHI'97 Human Factors in Computing Systems (March 22-27, Atlanta, GA), ACM Press, New York, 1997.

2. Stifelman, L.J., Arons, B., Schmandt, C. and Hulteen, E.A., *VoiceNotes: A speech interface for a hand-held voice notetaker*. In INTERCHI '93, pp. 179-186. ACM, 1993.

3. Bolt, R. A., *Put-that-there: voice and gesture at the graphics interface*. Computer Graphics, 14(3): pp. 262-270, 1980.

4. Koons, D., Sparrell, C. and Thórisson, K., *Integrating simultaneous input from speech, gaze and hand gestures*, Intelligent Multimedia Interfaces, ed. by M. Maybury, MIT Press: Cambridge, MA, pp. 257-276, 1993.

5. Cohen, P. R., Johnston, M., McGee, D., Smith, I., Oviatt, S., Pittman, J., Chen,L., and Clow, J., *QuickSet: Multimodal interaction for simulation set-up and control*. Proceedings of the Fifth Applied Natural Language Processing meeting. Association for Computational Linguistics: Washington, D.C., March, 1997.

6. Thórisson, K. R., *Layered modular action control for communicative humanoids*. Computer Animation '97, Geneva, Switzerland June 5-6, 1997.

7. Stallard, D., Bers, J., Barclay, C., *The GTNPhone Dialog System*. Proceedings of BNTUW98, Lansdowne, VA, February, 1998.